A Work Project presented as part of the requirements for the Award of a Master's degree in Management from the Nova School of Business and Economics.

OPTIMIZING ROUTES USING THE VEHICLE ROUTING PROBLEM

SOUFIANE BENMANSOUR

Work project carried out under the supervision of:
Dr. Manuel Pedro Baganha

The 20th of May

Abstract

With the emergence of e-commerce and its significant growth during the COVID-19 pandemic, more people aim to ship their products as fast as possible. More organizations such as DHL and USPS invest vast amounts of money in minimizing transportation costs while finding the shortest route that the delivery man should take to reach different destinations. My Work Project will evolve around studying the Vehicle Routing Problem that uses different algorithms to find the shortest route depending on various constraints. This project will include four parts: a literature review expressing an overview of the vehicle routing issue; then a state of the art of algorithm to understand the meaning behind the main algorithms. Finally, build a code (coding platform) using the algorithms discussed previously to solve a situation for the traveling salesman person and VRP.

Keywords

Vehicle Routing Problem, Traveling Salesman Problem, Route Optimization, Heuristic algorithms, logistics

Acknowledgment

- To **God**: For every good thing He offered me and the willingness to accomplish this Work Project
- To my **Parents** and **Family**: For their kindness, their big hearts, and their prayers.
- To my Professor **Dr. Manuel Baganha**: For guiding me throughout this whole journey

This work used infrastructure and resources funded by Fundação para a Ciência e a Tecnologia (UID/ECO/00124/2013, UID/ECO/00124/2019 and Social Sciences DataLab, Project 22209), POR Lisboa (LISBOA-01-0145-FEDER-007722 and Social Sciences DataLab, Project 22209) and POR Norte (Social Sciences DataLab, Project 22209

Table of Contents

I. Introduction

The issue of the VRP was characterized in 1959 by George Dantziget and John Ramser (Dantzig and Ramser 1959). This issue consists of optimizing routes of a bunch of vehicles to serve a group of clients. This problem has been studied for quite a while because its applications vary and for its extraordinary difficulty.

This issue is widespread in real life, which is why numerous variations of this topic have been made and considered to display these circumstances as precisely as possible. For instance, one of its most common variants is the Capacitated Vehicle Routing Problem, in which the goods should not surpass a specific capacity. In many cases, the situations studied consist of a mix of these variations, bringing about an exceptionally huge number of various issues for which explicit calculations have been created.

Before proceeding with the Vehicle Routing Problem, it is necessary to understand one particular case: the traveling Salesman Person or TSP; it is a simplified case in which only one vehicle can distribute goods. Understanding the algorithm behind it can explain how other complicated algorithms work. Therefore, this work project will focus first on solving TSP using a programming language interface.

There are various ways to deal with VRP and its numerous variations. A portion of these methodologies depends on altering existing results, for example, using past research. Others rely on creature practices; for example, researchers could generate algorithms depending on how ants were behaving. This was called the ant colony optimization algorithm (Bell and McMullen 2004; Paessens 1988; Bullnheimer, Hartl, and Strauss 1999). This paper aims to examine the degree to which these various techniques can be used together to discover the answer for the VRP.

Lastly, there will be an implementation of those findings in an organization that can be either physical or virtual and investigates the effect of this execution on this particular organization.

II. Literature review

In this part, the Vehicle Routing Problem overview will be given with its different types and components.

2.1. Vehicle Routing Problems

Being a non-deterministic polynomial problem or NP-Hard problem means that up until now, there is no efficient solution to solve this problem. This is the case for VRP that uses heuristic algorithms to find an approximate solution if the number of customers and constraints is high. The purpose behind it is to look for the best and optimal way to reach clients from a starting point (Depot) and under many constraints (Gilbert Laporte 1992). At first, oil companies mainly used it as they needed to deliver their product to gas stations every day, and the only constraint was to minimize the distance; it was called "The Truck Dispatching Problem." After five years of its introduction, Clarke and Wright used it as a linear optimization problem to evolve step by step until it became the "Vehicle Routing Problem" (Gilbert Laporte 1992).

2.2. Definition

The broad definition of VRP was introduced as follows: "A set of vehicles K have to start their trajectories from a starting point called Depot and a set of N customers that need to be served by the minimum number of cars. The problem is considered as a graph where customers are nodes and routes are vertices. Each path represents a set of customers that are served by one vehicle. The objective is to construct a set of routes that minimize the total distance/cost under the following constraints (Gilbert Laporte 1992):

- A vehicle m could not visit a customer more than once and should visit all assigned ones
- Departure and return should be at the exact location (Depot)
- The total capacity of the used vehicle is not violated.

2.3. VRP Components

To understand the concept of the VRP, it is necessary to know its main components that are necessary to facilitate the process of knowing the type and the solution of the algorithm used. The main constituents are:

- Depot: It is the starting and ending point of all the vehicles. Goods are carried from the Depot to the customers or vice versa. In most VRP cases, it is assumed that there is only one Depot.
- Road Network: It represents the base of VRP. It consists of the road connection that connects the Depot to the customer or the customers by themselves. The road network is shown mainly by the time it takes to cross each road or the distance/cost, and it can be either direct or undirect. In the undirected case, the vehicle can go both ways, while for the directed, the car can only go in one way. In addition to that, there is the location of each customer plus the Depot, which is the origin. In every VRP, the locations are represented as the vertex while the arcs are the road connections.
- Vehicle: In the case of VRP, there should be at least two vehicles. Having one car makes the problem more manageable, and it becomes another problem defined as the traveling salesman problem. Many constraints need to be considered while choosing a vehicle. Typical constraints are the loading capacity, the cost (fixed and variable), and the duration that refers to the timing that a vehicle can travel during a specific period.
- Operational Objectives: It depends on each company and what they want to minimize. They can choose between multi-objective or single objective. Some typical goals consist of reducing the number of trucks, or minimizing the traveling distance, or sometimes minimizing costs. In the case of multiple objectives, companies choose two or more purposes and combine them to get a single optimized solution for both.

- Constraint: Some constraints cannot be broken in all vehicle routing problems; for example, the demand cannot exceed the vehicle's capacity, the distance that can be traveled by the vehicle, and each customer cannot be visited more than once.

2.4. Variants of VRP

In a research done by Nathalie De Jaegere, she classified different types of vehicle routing problems according to the number of recurrences across all the papers that talk about it. In most articles, the one with the highest relative presence was the Capacitated Vehicle Routing Problem with a percentage of 88.89%. Then comes the Vehicle Routing Problem with Time Window with only a relative existence of 39.58% (Braekers, Ramaekers, and Van Nieuwenhuyse 2016). Below are some variants of VRPs:

- CVRP (Capacitated): As it was mentioned before, this is the most used and common vehicle routing problem. It consists of having a set of identical vehicles, meaning that they have the same capacity and speed. In this problem, customers, routes, and demand are all known, but the number of vehicles is unknown. It should be calculated considering the maximum capacity of the van and the need of each client. All the automobiles start from the same point, which is the Depot. Every customer must be visited only once, and all the demands must be met in that visit. Minimizing the total cost and reducing the number of vehicles driving off from the Depot is the main objective (Braekers, Ramaekers, and Van Nieuwenhuyse 2016).
- VRPTW (Time Window): This variant consists of restricting the timing in which the client should be served. It is divided into two divisions (Braekers, Ramaekers, and Van Nieuwenhuyse 2016):
    - Hard time window: vehicles can't deliver before or after the time interval. If the van comes before the timing, it had to wait, while in the case it is after the time interval, the result is considered invalid.

- Soft time window: the clients can accept the merchandise in this case but with imposing fines for the delivery company.

- VRPSD (Split Deliveries): In some cases, customers can be visited more than once, especially if the demand is higher than the vehicle's capacity or to reduce the number of vehicles. For example, if a customer has a particular need of 40 pieces and two vehicles can split this load, instead of using a new vehicle, the company can use those free spaces (Braekers, Ramaekers, and Van Nieuwenhuyse 2016).
- DVRP (Dynamic): Dynamic VRP is one of the most complex problems to solve. Its difficulty comes from the fact that there isn't enough information about the departure of the vehicle. They can change while the car is visiting the customers. For example, the driver can receive new information about a new client that needs to be visited to load or discharge merchandise (Braekers, Ramaekers, and Van Nieuwenhuyse 2016).
- There are many other variants of VRP (appendix 1). As mentioned before, it is possible to combine them depending on the company's need (multi-objectives).

III. State of Art of the Algorithms

This chapter will discuss the different algorithms used to solve diverse problems. These solutions were developed during this half-century and can be divided into two categories: The first category contains the exact algorithms that solve exact problems; the Second category contains the heuristic algorithms, and they can be either classical or metaheuristics.

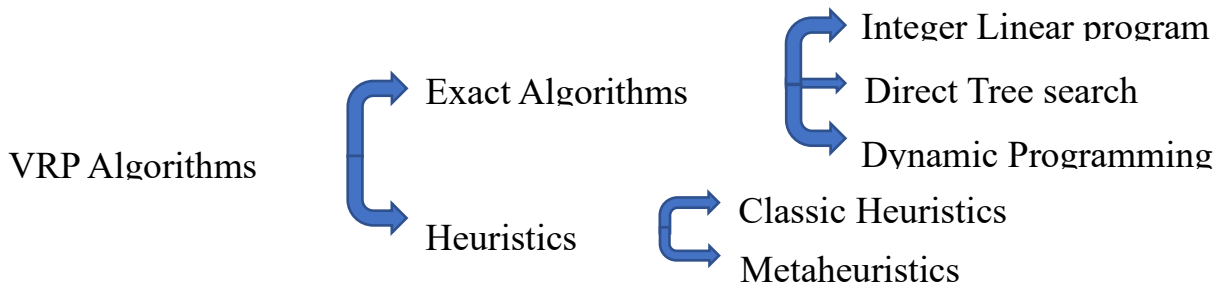


Figure 1: Different types of Algorithms

3.1. General formula

Before moving to discuss the algorithms, it is necessary to present a general formula for the VRP. This formula represents the basic one in which all the criteria that were discussed in the definition should be met. The mathematical model is as follows: "G = (V, A) where V = {0, 1, . . . , n} represents the vertex while A= {(i, j) : i, j ∈ V, i ≠ j} represents the arc." The Depot is always the Vertex n° 0, and every vehicle m (all of them with same capacity Q) should depart from it. Every customer i that belongs to V\{0} should have a non-negative demand and should always be less than the actual capacity of the vehicle. The constraint C_{ij} Should be defined on A. The type of road network being either directed or undirected determine if C_{ij} is symmetric or not. In case $C_{ij} = C_{ji}$ for all i, j; then the graph is undirected, and if one of them breaks the rule, then the graph is directed (Gilbert Laporte 2007).

Min $$\sum_{i \in V} \sum_{j \in V} C_{ij}\, x_{ij} \quad (1)$$

Subjected to

$$\sum_{i \in V} x_{ij} = 1 \quad \forall j \in V \setminus \{0\} \quad (2)$$

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i \in V \setminus \{0\} \quad (3)$$

$$\sum_{i \in V} x_{i0} = K \quad (4)$$

$$\sum_{j \in V} x_{0j} = K \quad (5)$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S),\ \forall S \subseteq V \setminus \{0\}, S \neq \emptyset \quad (6)$$

$$x_{ij} \in \{0,1\} \quad \forall i, j \in V \quad (7)$$

Those definitions were set by VRP founders Dantzig and John

(1) x_{ij} can be either 1 or 0, which means that the vehicle goes through this road (from i to j) or not. C_{ij} is the constraint that is most of the time either the cost, the time, or the distance. Overall,

this is the general formula that needs to be minimized; it should return a non-zero value subjected to the constraint that comes after it.

(2) (3) This formula makes sure that customers must be visited only once

(4) (5) This formula verifies that the total number leaving and returning to the Depot is precisely the same

(6) Ensures a connection between the Depot and the customers or the customers by themselves. It also verifies that the total demand for those specific clients that the vehicle will visit doesn't exceed the full capacity.

(7) Integrality constraint: make sure that x_{ij} return either 1 or 0

Some companies cannot afford to have as many vehicles as needed; in this case, it should be added to the constraints with the maximum number of trucks that can operate. Otherwise, the number is set to be free.

3.2. Exact Algorithms

Exact algorithms are the methods that find the exact solution (the most optimal one). It can be done if the number of locations that need to be visited doesn't exceed 50, and sometimes even less if there are many constraints. Supercomputers will take years to go over all the possible routes, and that's why it represents only a tiny percentage of the algorithms used (appendix 2). There are three approaches to solve it:

- Integer linear programming: This method is widely used for exact algorithms. Some of those are Agarwal's algorithm and Desrosiers. The algorithm uses a binary solution (1/0, yes/no) to indicate whether the solution can be feasible or not. It uses inequalities to find the optimal solution (Gilbert Laporte and Nobert 1987).

- Dynamic programming method: This method simplifies the problem by breaking the problem into a set of sub-problems. It starts by solving each sub-problem and then adding it to a partial solution until an optimal solution is found. It uses the algorithm that is mainly deployed in the traveling salesman problem but with an additional constraint to connect all vehicles (Gilbert Laporte and Nobert 1987).
- Direct tree search method: Christofides and Eilon first used this method in 1969 to solve the VRP. It is one of the first methods used in this problem. It goes over all possible solutions by creating a branch and bound tree that shows all possible routes starting from a specific point (J. F. Cordeau et al. 2002).

3.3. Heuristics Algorithms

The meaning of heuristics is any approach that solve a specific problem by applying certain methods that won't guarantee the optimal solution, but at least a close result. The gap between the optimal solution and the heuristic one can vary depending on the time taken to run the program and if there are too many constraints. Sometimes, the percentage difference can go less than 1% if the algorithm is sophisticated and takes time to be executed. With the rapid advancement of technology, AI is now used to improve those heuristics that can be used later on as a metaheuristic base. It is necessary to study the heuristic algorithms; the exact methods are no longer enough for huge companies that need instant results while facing different problems. Those algorithms are not only used in VRP but also in various other areas. The first simple heuristics were used between 1960 and 1990 for standard procedures; they are now limited as more and more constraints are added, and therefore it was necessary to develop them (G. Laporte et al. 2000).

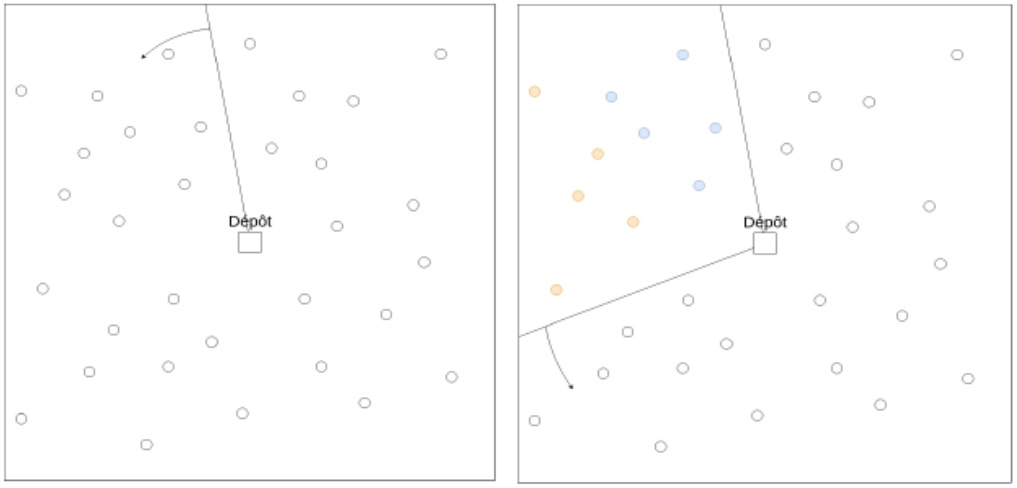
3.3.1. Classic heuristics

3.3.1.1. Heuristics of Construction

As its name describes, heuristics of construction start from scratch and gradually construct a solution close to the optimal one either in cost or distance. They rely on simple methods that humans can generally solve but in a short period. Some of them are as follow:

Clarke and Wright Heuristic: It was proposed in 1964 to solve the capacitated vehicle routing problem. It creates simple solutions and starts by optimizing each one until a good solution is found. The logic behind it works as follow: let's consider that each customer will be visited by one vehicle; this means that the number of customers n is equal to the number of vehicles and that the distance traveled by each vehicle is **2*d(D,i)**, so the total distance traveled by all vehicles is $2* \sum_i^n d(D,i)$. Let optimize then the solution and start by linking two customers to one single vehicle; this means that the distance saved is **s(i,j)= d(D,i) + d(D,j) – d(i,j).** This can only be done if visiting those two customers won't violate a constraint; for example, the total demand doesn't exceed the total capacity. The higher the s(i,j)'s value, the better. The program keeps linking customers until s(i,j) start increasing, and therefore it concludes that a good solution was found. This method doesn't consider the constraint of the maximum number of vehicles that a company can deploy. Some variations were introduced to consider other constraints, but the principle stays the same (Clarke and Wright 1964). This method will be discussed further in the next chapter.

Sweep algorithm: Wren and Holiday first used it for the capacitated VRP in 1972. It's a straightforward algorithm that puts all the locations into a map and starts scanning all the ones close to one another and with an amount of demand that shouldn't exceed the total capacity and then put them together. The extremity of the axis that scan the map starts from the Depot, as you can see in the figure:



The axis has an origin from depot and start scanning counterclockwise

The axis defining the customers that need to be visited with different colors

Figure 2: Sweep Algorithm Demonstration

Different colors were assigned to clients; whenever the total capacity of a vehicle is reached, the axis starts setting different colors. After scanning the whole map, the program begins solving each problem independently as a traveling salesman problem. Many disadvantages can be faced while using this method; for example, by starting the axis in a different position or changing the scan from counterclockwise to clockwise, a whole other solution might be generated. There are also many constraints that are not considered, but overall, this is one of the fastest methods to create a solution for VRP (Wren and Holliday 1972).

3.3.1.2. Route improvement heuristics

Improvement algorithms start by using solutions from other heuristics or just put a random solution initially; for example, it can start from a path that was found by using the sweep algorithm. The way it works is by making some changes either for routes or clients and see if this reduces the cost or the distance. If it's the case, the algorithm keeps it; if not, it returns to the previous solution. Two ways are possible to solve this problem: either using Intra route or

Inter route change. The first one consists of changing the client's order within the same path, while the latter involves changing a customer's positioning into another set of arcs.

2-opt Search: 2-opt is an intra-route method. It comes from the fact that for every arc, there can be a maximum of 2 changes. This means that if two customers rely upon an arc, the maximum changes that can be made is to change the direction that the vehicle can take after visiting the second client or the origin from where it comes while visiting the first client. After adjusting the arc, the cost/distance is calculated to see if there was a decrease or increase using this formula: (the cost/distance of removed edges) – (the cost/distance of added edges). If the formula returns a positive value, then the algorithm keeps it as this number is the amount of cost/distance saved; if not, it returns to the previous one. This algorithm is mainly used to remove the intersected arcs (G. Laporte et al. 2000).

The ejection chain is an Inter-route method. As its name mention, the ejection chain starts by choosing a specific set and makes it go through a change either by having a new position or changing its value. In VRP, the only change that can be made is by swapping its position into another route. The algorithm starts then by comparing if the added client reduces the total cost or it should be rejected until a solution is found (GLOVER 1992).

3.4. Metaheuristic Algorithms

In software engineering and numerical advancement, a metaheuristic is a more significant level technique or heuristic intended to discover, create, or select a search algorithm calculation that may adequately answer an enhancement issue, particularly with fragmented or blemished data or restricted calculation limit. In general, those algorithms sometimes use the heuristic methods as a base.

Metaheuristics test many results that are too huge and can't be examined. Metaheuristic may make not many hypotheses about the issue being tackled; thus, they might be used to solve

different issues and not only VRP. The metaheuristic can frequently discover great resolutions with less computational exertion and streamlining calculations; accordingly, they are helpful methodologies for advancement issues.

In general, metaheuristics are characterized as follow (Blum and Roli 2003):

- Metaheuristics use algorithms to guide search processes
- They are the most advanced algorithms when it comes to finding near-optimal solutions efficiently
- They can range from a simple search to complex heuristics that uses learning processes
- They are non-deterministic and guarantee approximate solutions
- Metaheuristics are problem independent, which means that the same algorithm can solve different problems

There are three categories of metaheuristics: 1. Local search, 2. Population search, 3. Neural network (not very popular).

3.4.1. Local Search

Local search methods investigate the current arrangement of the neighborhood and move every repetition of that arrangement to the same area. Various sorts of strategies exist inside this classification to characterize and investigate neighborhoods. The first one that suggested using local search was Osman in 1993 (Osman 1993). He talked about simulated annealing and tabu search for VRP. This later became one of the most studied in VRP during those last two decades (Jean Frangois Cordeau and Laporte 2005).

Tabu search was introduced by Fred Glover in 1984; it uses an approach that consists of remembering old properties of already done solutions and ensuring that they will not be treated again. Those solutions are stored in a list called 'tabu list.' This method avoids falling into a loop of the exact solutions that most other programs fall into. This metaheuristic approach is called 'Taburoute' (Gendreau, Hertz, and Laporte 1994). This algorithm works by constantly

disconnecting a vertex in a set of routes and then implementing it into another route. This algorithm has a different aspect, namely (Meliani et al. 2019):

- Search Space: it's the whole set of solutions that can be visited during the search.
- Neighborhood structure: It's the whole set of operations applied to the solutions to improve the quality of results.
- Intensification: this concept consists of reinforcing the search into certain parts of the search space to find the local optimum in that region.
- Diversification: It's the opposite of intensification. This method consists of making sure that the algorithm explores new regions to find the global optimum of the search space and not keep going around in a particular area.

3.4.2. Population Search

The population search mimics the interaction of nature, such as animals. It works by looking at how nature works, how animals move in specific ways, and try to imitate their interaction by finding ways to create algorithms that work in the same way hoping that it might be efficient.

Two researchers used this method to solve problems for the classical VRP. They have shown in their paper that this method can indeed be very efficient compared to other heuristics in terms of the quality and the time taken to compute and find a suitable solution (Baker and Ayechew 2003).

One of the methods that attracted many researchers was Ant Colony Optimization. It was developed 15 years ago by Chen & Ting in 2006 for essential use, such as the capacitated VRP (Chen and Ting 2006). Some other complex versions were developed to include more constraints; one of them is the multiple ant colony used for the vehicle routing problem with the time window. The first method consists of optimizing the total vehicles, as for the second

one, it reduces the total distance. This method was inspired by the conduct of ants while looking for food in nature. By noticing their behavior, the scientists found that they would, in general, utilize the briefest accessible way to their food source regardless of their restricted intellectual capacities. This is clarified as follows:

- Scout ants move randomly in different paths while looking for food
- They go back to their nest after finding food, but while returning, they leave behind them a trail of pheromones
- Other ants will be attracted by the pheromones when returning to the nest and will release their pheromones in turn
- If more than one pheromone track is accessible, the briefest track will, in general, be followed as more ants will take it in a short period
- Other pheromone paths will, in general, vanish as fewer and fewer ants will follow it

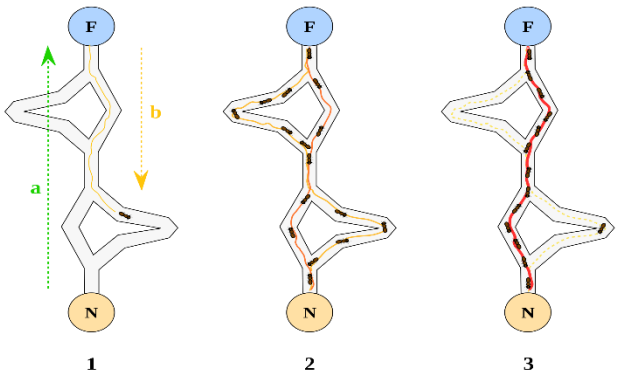


Figure 3: Example of how ant colony optimization works

The first drawing on the left shows an ant that leaves behind it a pheromone after finding food F. The same happens for other ants (drawing 2) until one optimal path has more pheromones, and therefore, more ants will follow it (path 3) (James, n.d.).

IV. Methodology

The following chapters will start by solving two different problems. The first one will consist of solving a Traveling Salesman Problem (TSP), while the second one will go deeper into solving a Capacitated Vehicle Routing Problem. In every case, all the components of TSP and VRP should be present (Depot, road network, vehicle, and operational objectives). The resolution approach considered in those two cases is based on Clark and Wright's constructive heuristics and local search; a MATLAB and VRP Spreadsheet solver is introduced at the end.

V. Solving a TSP case study

As it was mentioned in the methodology, this chapter will focus on solving a real-life case study and try to compare the different results that were found with the solution that is already used. One of the essential stations in Lisbon is "Cais Sodré"; it is where the train station intersects with the bus and metro station. The line chosen is 22B (Moovit 2020); the bus starts from Cais Sodré and visits 12 stations to return to Cais Sodré. First, it is necessary to find the coordinates of every station as they will be included in the coding to find the distances between each location (appendix 3).

This chapter will be divided into four parts; the first three parts will solve the problem using different algorithms and programs; those algorithms include Clark and Wright and Local Search. The programs used will be JAVA and MATLAB. The last chapter will compare different results.

5.1. Clarke and Wright Algorithm

As discussed in (3.3.1.1.), Clark and Wright's algorithm is simple to use. It follows this procedure to find an optimal solution:

1. Calculate the saving-cost between each location (i,j)
2. Put all those calculation into a list called "saving list" and order them

3. Start going over the list from the top
4. For every $s_{i,j,}$ (i,j), the following cases are considered:
    a) If i or j are not part of a route, the algorithm creates a new way that includes i and j
    b) If a customer i was included in a particular path, but it isn't adjacent with the Depot (not interior), then the path (i,j) is included in the route as long as it doesn't violate a constraint; otherwise, a new way will be included.
    c) If point i and point j each of them is part of a different route and they aren't adjacent with the Depot, then both routes will be joined by (i,j)
5. In case the saving-list is not drained, the algorithm updates the least one and repeats number four
6. In case a customer is not included in a route after the repetition, then a vehicle is assigned to that specific client.

In this case, there will be only one route and one vehicle as this is a traveling salesman problem, this makes the problem simpler. After writing the algorithm into JAVA and including every coordinate into it (appendix 4), the final route is a follow:

cln : **R. S. Paulo (Bica)** cln : **R. Alecrim** cln : **Pç. Luis Camões (B.º Alto)** cln : **R. S. Bento / Cç. Estrela** cln : **Príncipe Real** cln : **Calhariz (Bica)** cln : **R. Rosa** cln : **R. Rosa / Trav. S. Pedro** cln : **R. Palmeira** cln : **Pç. Flores** cln : **Palácio S. Bento (Jardim)** cln : **Conde Barão**

5.2. Local Search Algorithm

Basic local search will be used in this case, it uses the same logic as tabu search that was discussed in past chapters but without having a tabu-list that will store past finding in order not to fall into a loop. The algorithm was implemented using Java (appendix 5) and the results were as follow:

cln : **R. Alecrim** cln : **Pç. Luis Camões (B.º Alto)** cln : **Calhariz (Bica)** cln : **R. Rosa** cln : **R. Rosa / Trav. S. Pedro** cln : **Príncipe Real** cln : **R. Palmeira** cln : **Pç. Flores** cln : **Palácio S. Bento (Jardim)** cln : **R. S. Bento / Cç. Estrela** cln : **Conde Barão** cln : **R. S. Paulo (Bica)**

5.3. MATLAB for the Graphic Solution

MATLAB is a programming language that is popular among mathematicians and scientists. It is used chiefly to manipulate matrices, plot graphs, and implement algorithms. For the Vehicle Routing problem, it is the perfect tool to help generate graphs while solving algorithms to have an idea of the shape of the route. To have an accurate solution, it was essential to calculate the real distances between each location manually (appendix 6) because MATLAB uses direct lines automatically to calculate distances. The algorithm used here is exact; it starts by developing a random solution and then goes over all possible routes and ranks them accordingly. The number of possible solutions if all coordinates are connected in a TSP is always (n-1)!. In this case, and since there are 12 destinations (excluding the starting and endpoint "Cais Sodré"), the total number of solutions is 39.6 million solutions. This can take a while to be calculated by a computer, and in case there are more than 25 destinations, supercomputers can take more than 2.46 years to find all possible solutions (appendix 1).

After implementing the algorithm and coordinates into MATLAB, the graphic solution found was as follow:

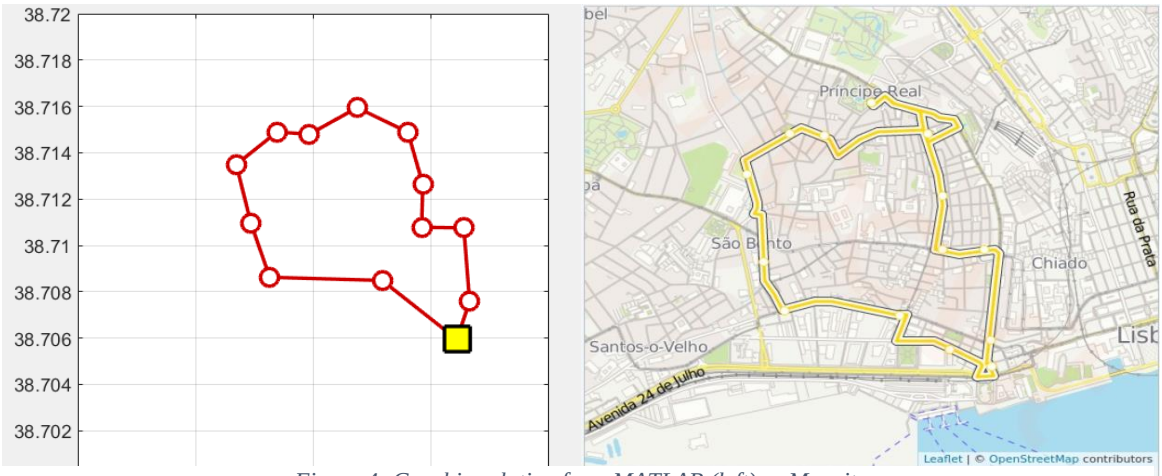


Figure 4: Graphic solution from MATLAB (left) vs Moovit

5.4. Comparison of Results

The solutions stated above show a difference between the different results; for example, when comparing the MATLAB solution (Exact) to the Local Search solution (metaheuristic), there is a slight difference in the client's order that should be visited. The order of Principe Real and R. Rosa/ Trav. S. Pedro has changed in LC. For Clark and Wright algorithm, the order of stations is very different than the exact algorithm. This can be normal as the CW is a heuristic solution which means that it will not necessarily find the optimal solution but rather a close one. The CW algorithm is usually used to construct an initial solution for developed metaheuristic algorithms.

The actual route taken by bus number 22B (appendix x) is the same as the one found using MATLAB; therefore, the bus is taking the optimal path, and no changes can be made. The same procedure can be done for all bus tours to optimize both the distance and the fuel consumed.

VI. Concrete Application of VRP to Promotherme Environnement

6.1. Overview

Promotherme Environnement is a water treatment company based in Casablanca, the largest city in Morocco. It was founded in 1998 by Mr. Said Bouselha. As of 2015, it was considered the leading supplier and manufacturer of Boilers in Morocco (Promotherme 2013). They were able to be the supplier of huge and multinational companies such as International Paper, Total, Shell and Vivo Lubricant, Coca-Cola… In 2010, they started delivering everything related to water treatment (boilers, water purifiers, reservoirs…) to the whole country, especially Casablanca, Marrakech, Rabat, Tangier Agadir, Dakhla, Laayoune, and Mohammedia, as those are the heart of the Moroccan economy.

6.2. Problem description

To be competitive nationally, Promotherme located its Depot in Ain-Sebaa, a district of Casablanca situated near the highway, to ease the import and export of materials. With the increase of customer demand, it is becoming more and more difficult to supply all material on time. Therefore, the company's chief executive officer was thinking about increasing the number of trucks or outsourcing the delivery process to satisfy his clients and be always ready for the increase of demand. Supposing they choose to go with the first suggestion, one problem they were facing is knowing the optimal number of trucks they should purchase. The clients are randomly distributed nationally, particularly in Casablanca, where more than 35% of the clients are located.

6.3. Analysis of the Company

The demand can sometimes go up to 120 m^3 in Casablanca and 180 m^3 nationally every two to three days. The volume of one industrial steam boiler for oil and gas firing can sometimes reach 114 m^3. Orders are subjected to an increase in the following years with the new vision of the company to start shipping its product abroad. The company schedule the national delivery on Mondays and Thursdays, while for Casablanca, the delivery is made on Tuesdays, Wednesdays, and sometimes on Fridays on special occasions. The first problem was to forecast the demand as it is random; one client can order a huge boiler that can fit in one truck, while other times, one truck can load the request of 7 clients. For this, using demand forecasting with regression was needed. The company demand will be computed in m^3 to avoid diversity in products. Because of COVID, it was required to choose a year where the demand was regular to avoid having false forecasting. The period chosen was from January 2019 till February 2020. The raw data contained only the general demand of each week divided between Casablanca and the other cities. It was necessary to sit with the logistic manager and approximate the day when the order was the highest. As this was backdated (two years ago), an approximation was given

to move on with the project (confidential information, so the demand table will not be included in the report). The conclusion was that by the year 2025, the demand would increase by approximately 35%, and the cities that will need more resources are Casablanca, Tangier, Dakhla.

The executives chose to apply the VRP only in Casablanca. Distribution of national demand is clear, and they already know how to allocate the demand for every client. To know the number of trucks needed, johit was essential to take one of the highest demands received in Casablanca and implement it in the VRP Spreadsheet to calculate the optimal distance and the optimal number of vehicles.

6.4. VRP Spreadsheet Slover

The VRP spreadsheet Slover is a program that is implemented into excel. It uses metaheuristic algorithms (the Large Neighborhood Search (LNS)) to find an approximate optimal route. It is very flexible when using it. It gives its users the ability to choose the number of depots, the route type, the average vehicle speed, time window, warm start, CPU running time, and finally, the distance computation method (appendix 7). Users can choose to use real distances by having a Bing Map Key that allows the program to generate Bing distances rather than using the straight-line method. It is also made for daily use and generates graphic solutions to have a whole picture of the different routes and each client's positioning.

6.5. Data Implementation

After analyzing the demand that was received on a weekly basis, it was concluded that boilers exceeding 45 m^3 will not be included; those cisterns need special trucks. Another reason is that the company plans to buy Truck Wingbox for nationwide delivery and truck colt diesel Dobel box within Casablanca and near cities. Those trucks do not exceed 50 m^3 and 12 m^3. Before implementing data, it was necessary to study all the constraints faced while loading the

merchandise. The first one was that there will always be free spaces between every piece of merchandise that cannot be filled. While for the second constraint was that some parts should always be loaded on the top because they are fragile. For that, the capacity assumed for each truck should be reduced by 30%.

The current situation is assumed to be before Covid-19 (the beginning of 2020). It is also important to note that the executives decided that within Casablanca, trucks can not return to Depot multiple times. The time it takes to load and unload the demand can be very important. Also, sometimes clients ask for some special request that might elongate unloading time. For confidential reasons, the locations will be approximative, and no customer name will be revealed. The day that was chosen within Casablanca was the 9th of January, where the demand reached 84 m^3 from different customers (Appendix 8). The problem solving started first by enumerating the customers, choosing an average vehicle speed, and then numbering the vehicles. The next step was to locate the customers (approximate) and then compute the distance between the client themselves and the Depot (appendix 8). Then comes setting the vehicle worksheet in which different information about the vehicle is set (appendix 9). The number of vehicles included was high to set the number as free. The maximum capacity of every vehicle was reduced to 8.4 m^3even though the maximum capacity of the vehicle is 12 m^3. After implementing the data, it was observed that some orders exceed the maximum capacity, and this will make an unfeasible solution; for this reason, two Wingbox trucks were added to resolve this issue.

The solution generated (appendix 10) shows that the number of Truck Wingbox was enough to load all the demand that exceeded the capacity of the standard truck. As for the truck colt diesel Dobel box, the minimum number of trucks that can meet the demand is seven, as indicated:

Table 1: Customers that need to be visited by each vehicle

Truck type	Location Name	Distance travelled	Delivery amount	Load	Total Load
V1 (Dobel box)	Customer 11	17.296	4.1	0	**4.1**
V2 (Dobel box)	Customer 18	8.78	2.6	4.5	**7.1**
	Customer 2	10.8	0.5	4	
	Customer 1	11.705	4	0	
V3 (Dobel box)	Customer 7	12.733	6.4	1.2	**7.6**
	Customer 5	24.775	1.2	0	
V4 (Dobel box)	Customer 15	10.088	4.5	2.5	**7**
	Customer 4	17.936	1.5	1	
	Customer 3	21.228	1	0	
V5 (Dobel box)	Customer 17	47.215	2.8	5.2	**8**
	Customer 8	79.639	5.2	0	
V6 (Dobel box)	Customer 6	27.584	8	0	**8**
V7 (Dobel box)	Customer 12	26.157	6.3	1.8	**8.1**
	Customer 13	34.459	1.8	0	
V13 (Wingbox)	Customer 10	17.415	16	0.6	**16.6**
	Customer 9	21.137	0.6	0	
V14 (Wingbox)	Customer 16	40.55	9.3	8.7	**18**
	Customer 14	121.038	8.7	0	

6.6. Suggestions and Limitations of Obtained Results

The result shows that buying 7 Dobel boxes and 2 Wingbox is enough to meet the demand in Casablanca. Since Promotherme already has 4 Dobel Box, they will only need three others. This is the result of a metaheuristic algorithm, which means that other efficient solutions might exist. Furthermore, the table shows that most of the vehicles are fully filled (with the exception of vehicle 1); this implies that the company needs to invest more if they are forecasting that the demand will rise by 35% in the upcoming years. Some suggestions might include buying other trucks rather than the Dobel, but the company will have to look for another partner instead of their main truck supplier. They can also consider that the demand is increasing; therefore, they shouldn't focus only on the actual need. Finally, they should outsource some of their demand if a client does not exceed a certain threshold.

For the limitations, only one constraint was included, which is the capacity. In real life, there are many other constraints, such as the time windows and split delivery. These variants can be added in future situations, but it can be tough to implement them as Promotherme should consider too many other assumptions. Another limitation lies in the software; a small amount of memory was used to find the optimal solution. This is because the laptop used was not very powerful; this implies that different results can be found but not very far from one another. The addition of the memory can avoid cycling and improving the performance of the approach. Finally, the program was not tested on large instances (50 clients), so there is no guarantee if the program might work or not if more customers are added in the future. Since there were only 18 customers in total, this program can be used for many upcoming years.

VII. Conclusion

This work project would not have been possible without the extreme determination during this whole semester. The VRP is one of the most complex issues faced by everyone working in the supply chain and logistics. The main purpose of this problem is to find the most optimal solution for a set of routes and vehicles while including many constraints that make the situation even harder. Unfortunately, the specified period for this work project was not enough to tackle all aspects of the VRP or go deeper into its variants. In general, it was an exciting and enriching experience through the implementation of what was learned into different cases. One of the many challenges faced during this semester was the use of software programs. It was necessary to include a computer scientist to implement the algorithms into different platforms, namely MATLAB and JAVA. The many meetings that were set with PROMOTHERME employees were also challenging since they did not have prior knowledge about this subject. It was necessary to make everything done in two weeks. Finally, many thanks and appreciation to whomever contributed even by a little to make this Work Project feasible, THE VEHICLE ROUTING PROBLEM.

References

Baker, Barrie M., and M. A. Ayechew. 2003. “A Genetic Algorithm for the Vehicle Routing Problem.” *Computers and Operations Research* 30 (5). https://doi.org/10.1016/S0305-0548(02)00051-5. Page 15

Bell, John E., and Patrick R. McMullen. 2004. “Ant Colony Optimization Techniques for the Vehicle Routing Problem.” *Advanced Engineering Informatics* 18 (1). https://doi.org/10.1016/j.aei.2004.07.001. Page 3

Blum, Christian, and Andrea Roli. 2003. “Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison.” *ACM Computing Surveys*. https://doi.org/10.1145/937503.937505. Page 14

Braekers, Kris, Katrien Ramaekers, and Inneke Van Nieuwenhuyse. 2016. “The Vehicle Routing Problem: State of the Art Classification and Review.” *Computers and Industrial Engineering*. https://doi.org/10.1016/j.cie.2015.12.007. Page 7

Bullnheimer, Bernd, Richard F. Hartl, and Christine Strauss. 1999. “An Improved Ant System Algorithm for the Vehicle Routing Problem.” *Annals of Operations Research* 89. https://doi.org/10.1023/a:1018940026670. Page 3

Chen, Chia Ho, and Ching Jung Ting. 2006. “An Improved Ant Colony System Algorithm for the Vehicle Routing Problem.” *Journal of the Chinese Institute of Industrial Engineers* 23 (2). https://doi.org/10.1080/10170660609509001. Page 15

Clarke, G., and J. W. Wright. 1964. “Scheduling of Vehicles from a Central Depot to a Number of Delivery Points.” *Operations Research* 12 (4). https://doi.org/10.1287/opre.12.4.568. Page 11

Cordeau, J. F., M. Gendreau, G. Laporte, J. Y. Potvin, and F. Semet. 2002. “A Guide to Vehicle Routing Heuristics.” *Journal of the Operational Research Society* 53 (5). https://doi.org/10.1057/palgrave.jors.2601319. Page 10

Cordeau, Jean Frangois, and Gilbert Laporte. 2005. “Tabu Search Heuristics for the Vehicle Routing Problem.” *Operations Research/ Computer Science Interfaces Series* 30. https://doi.org/10.1007/0-387-23667-8_6. Page 14

Dantzig, G. B., and J. H. Ramser. 1959. “The Truck Dispatching Problem.” *Management Science* 6 (1). https://doi.org/10.1287/mnsc.6.1.80. Page 3

Gendreau, Michel, Alain Hertz, and Gilbert Laporte. 1994. “Tabu Search Heuristic for the Vehicle Routing Problem.” *Management Science* 40 (10). https://doi.org/10.1287/mnsc.40.10.1276.Page 14

GLOVER, FRED. 1992. “NEW EJECTION CHAIN AND ALTERNATING PATH METHODS FOR TRAVELING SALESMAN PROBLEMS.” In *Computer Science and Operations Research*. https://doi.org/10.1016/b978-0-08-040806-4.50037-x. Page 13

James, Cecil. n.d. “Ant Colony.” Pinterest. https://www.pinterest.nz/pin/406872147564794743/. Page 16

Laporte, G., M. Gendreau, J. Y. Potvin, and F. Semet. 2000. “Classical and Modern Heuristics for the Vehicle Routing Problem.” *International Transactions in Operational Research* 7 (4–5). https://doi.org/10.1111/j.1475-3995.2000.tb00200.x. Page 13

Laporte, Gilbert. 1992. “The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms.” *European Journal of Operational Research* 59 (3). https://doi.org/10.1016/0377-2217(92)90192-C. Page 4

———. 2007. “What You Should Know about the Vehicle Routing Problem.” *Naval Research Logistics* 54 (8). https://doi.org/10.1002/nav.20261. Page 8

Laporte, Gilbert, and Yves Nobert. 1987. “Exact Algorithms for the Vehicle Routing Problem.” *North-Holland Mathematics Studies* 132 (C). https://doi.org/10.1016/S0304-0208(08)73235-3. Page 10

Meliani, Youssef, Yasmina Hani, Sâad Lissane Elhaq, and Abderrahman El Mhamedi. 2019. "A Developed Tabu Search Algorithm for Heterogeneous Fleet Vehicle Routing Problem." *IFAC-PapersOnLine* 52 (13). https://doi.org/10.1016/j.ifacol.2019.11.334. Page 15

Moovit. 2020. "Cais Sodré - Circulação Príncipe Real." Cais Sodré - Circulação Príncipe Real. 2020. https://moovitapp.com/index/en/public_transit-line-22B-Lisboa-2460-1550170-17564007-0. Page 17

Osman, Ibrahim Hassan. 1993. "Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem." *Annals of Operations Research* 41 (4). https://doi.org/10.1007/BF02023004. Page 14

Paessens, H. 1988. "The Savings Algorithm for the Vehicle Routing Problem." *European Journal of Operational Research* 34 (3). https://doi.org/10.1016/0377-2217(88)90154-3.

Promotherme. 2013. "Promotherme Website." 2013. http://www.promotherme.com/. Page 3

Wren, Anthony, and Alan Holliday. 1972. "COMPUTER SCHEDULING OF VEHICLES FROM ONE OR MORE DEPOTS TO A NUMBER OF DELIVERY POINTS." *Operational Research Quarterly* 23 (3). https://doi.org/10.1057/jors.1972.53. Page 20

Appendix

Appendix 1:

Table 2: Variants of VRP

Variant	Relative presence
CVRP (Capacitated)	88.89%
VRPTW (Time Window)	39.58%
HVRP (Heterogeneous)	18.75%
MDVRP	12.50%
VRPPB	11.81%
SDVRP	11.11%
DVRP (Dynamic)	10.42%
PVRP	9.72%
VRPSD	9.03%
VRRSPD	8.33%
OVRP	6.25%
TDVRP	4.86%
MCVRP	3.47%
CCVRP	2.08%

Appendix 2:

Table 3: Enumeration for different nodes

Number of vertices	Number of tours	Supercomputer enumeration time
5	24	< 1 second
10	362,880	< 1 second
15	87,178,291,200	< 1 second
20	121,645,100,408,832,000	12.165 seconds
25	620,448,401,733,239,000,000,000	2.46 years
30	8,841,761,993,739,700,000,000,000,000	> 42 million years

Appendix 3:

Table 4: coordinates of every station

Station	Latitude (y)	Longitude (x)
Cais Sodré	38.705946	-9.143902
R. Alecrim	38.707572	-9.143355
Pç. Luis Camões (B.º Alto)	38.71076	-9.143602
Calhariz (Bica)	38.710784	-9.145395
R. Rosa	38.712643	-9.145349
Príncipe Real	38.715947	-9.148136
R. Rosa / Trav. S. Pedro	38.714891	-9.145982
R. Palmeira	38.7148	-9.150173
Pç. Flores	38.714897	-9.15155
Palácio S. Bento (Jardim)	38.713498	-9.15328
R. S. Bento / Cç. Estrela	38.710962	-9.152661
Conde Barão	38.708625	-9.151857
R. S. Paulo (Bica)	38.708489	-9.147074
Cais Sodré (Centro Saúde)	38.705946	-9.143902

Appendix 4:

```
public List <Route> savingAlgo (List <Cmd2> ors) {
    List<SavingL> sl = calcSaving(ors);
    List<Route> rts = new ArrayList<~>();

    for (SavingL s:sl){
        int[] res = findclients(s.getCln1(),s.getCln2(),rts);
        creatRoute(res,s.getCln1(),s.getCln2(),rts);
    }
    return rts;
}
```

```java
public int[] findclients(Cmd2 cln1,Cmd2 cln2, List<Route> rts){
    int[] clnout ={-1,-1};
    for(int i=0;i<rts.size();i++){
        for (Cmd2 cmd : rts.get(i).getOrders()){
            if (cmd.getClient().equals(cln1.getClient())){
                clnout[0]=i;
            }
            if (cmd.getClient().equals(cln2.getClient())){
                clnout[1]=i;
            }
        }
    }return clnout;
}
```

```java
public void creatRoute(int [] res,Cmd2 cln1,Cmd2 cln2,List<Route> rts ){

    int i = res[1];
    int j = res[0];

    if (j == -1 && i == -1){
        Route rt = new Route(clients.get(0),clients.get(0));
        List<Cmd2> cls = new ArrayList<~>();
        cls.add(cln1);
        cls.add(cln2);
        rt.setOrders(cls);
        rts.add(rt);


    }else if (j == -1 && i != -1){
        if (rts.get(i).getOrders().get(0) == cln2){
            rts.get(i).getOrders().add( i: 0, cln1);
        }
        else if (rts.get(i).getOrders().get((rts.get(i).getOrders().size())-1) == cln2) {
                List<Cmd2> o = rts.get(i).getOrders();
                o.add(cln1);
                rts.get(i).setOrders(o);

        }
    }else if (j!=-1 && i==-1){
        if (rts.get(j).getOrders().get(0) == cln1){
            rts.get(j).getOrders().add( i: 0, cln2);
        }else if (rts.get(j).getOrders().get(rts.get(j).getOrders().size()-1).equals(cln1)){
            rts.get(j).getOrders().add(cln2);
        }
```

```java
public class HFVRP {

    public static void main(String[] args) {
        Algo algo = new Algo();
        algo.fun();
    }
}
```

Appendix 5:

```
public List<Route> localSearchProcedureSwapIntraRoute (List<Route> rts){
    List<Route> rtsFirst= cloneRoutes(rts);
    List<Route> rtsSwap= cloneRoutes(rts);
    List<Route> rtsSwapTwoOne= cloneRoutes(rts);
    List<Route> rtsInsertion= cloneRoutes(rts);
    List<Route> rtsInsertionTwo= cloneRoutes(rts);

    int x=0;
    int rand1 = (int)(Math.random() * (orders.size()));

    int u =0;
    int rand2 = (int)(Math.random() * (orders.size()));

    int v =0;
    int rand3 = (int)(Math.random() * (orders.size()));

    int b =0;
    int rand4 = (int)(Math.random() * (orders.size()));

    rtsSwap = swapProcedureIntraRoute(rtsSwap,rand1);

    rtsSwapTwoOne = exchangeTwoOneIntraRoute(rtsSwapTwoOne,rand2);

    rtsInsertion = insertionProcedureIntraRoute(rtsInsertion,rand3);

    rtsInsertionTwo = insertionProcedureTwoIntraRoute(rtsInsertionTwo,rand4);



    List<Route> minSol = minSol2(rtsInsertion,rtsSwap, rtsSwapTwoOne,rtsInsertionTwo);

    return minSol;
}
```

Appendix 6:

Table 5: Distance between each client

Station	Cais Sodré	Calhariz (Bica)	Conde Barão	Palácio S. Bento (Jardim)	Pç. Flores	Pç. Luis Camões (B.º Alto)	Príncipe Real	R. Alecrim	R. Palmeira	R. Rosa	R. Rosa / Trav. S. Pedro	R. S. Bento / Cç. Estrela	R. S. Paulo (Bica)
Cais Sodré	0	0.933	1.303	1.878	2.12	0.776	1.6	0.371	1.86	1.255	1.402	1.762	0.885
Calhariz (Bica)	0.755	0	0.864	0.981	1.03	0.323	0.773	0.559	1.16	0.224	0.481	0.679	0.772
Conde Barão	0.965	0.83	0	0.706	0.95	1.154	1.338	1.297	1.08	1.016	1.273	0.59	0.712
Palácio S. Bento (Jar	1.816	1.14	0.595	0	0.25	1.554	0.631	1.807	0.38	1.326	1.137	0.313	1.34
Pç. Flores	1.809	1.145	0.841	0.245	0	1.308	0.385	1.613	0.13	1.218	0.891	0.559	1.586
Pç. Luis Camões (B.º	0.648	0.156	1.021	1.138	1.18	0	0.93	0.452	1.32	0.381	0.638	0.836	0.776
Príncipe Real	1.423	0.759	1.315	0.641	0.4	0.922	0	1.227	0.26	0.832	0.505	1.13	1.551
R. Alecrim	0.195	0.561	1.08	1.693	1.94	0.404	1.228	0	1.49	0.883	1.03	1.577	0.662
R. Palmeira	1.677	1.013	0.87	0.377	0.13	1.176	0.253	1.481	0	1.086	0.759	0.685	1.492
R. Rosa	1.013	0.323	1.188	1.305	1.05	0.512	0.548	0.817	0.81	0	0.256	1.003	1.073
R. Rosa / Trav. S. Pe	1.322	0.884	1.548	1.042	0.8	0.821	0.291	1.126	0.56	0.731	0	1.363	1.45
R. S. Bento / Cç. Estr	1.351	0.826	0.281	0.987	1.12	1.15	1.504	1.386	1.25	1.012	1.269	0	1.098
R. S. Paulo (Bica)	0.522	1.248	0.418	1.124	1.37	1.18	1.756	0.775	1.5	1.434	1.691	1.008	0

Appendix 7

Table 6: VRP Solver Console

Sequence	Parameter	Value	Remarks
0.Interface	Language	English	Please refer to the manual for modifying the interface.
	Optional - Bing Maps Key	ApsJnANyZOr9hEax4Sdd8ZW_QWI-mRVnjBRPlLL7Yq4KgLyPzn-vbErx7eOUR6-F	You can get a free trial key at https://www.bingmapsportal.com/
1.Locations	Number of depots	1	[1,20]
	Number of customers	18	[5,200]
2.Distances	Distance computation method	Bing Maps driving distances (km)	Recommendation: Use 'postcode, country' format for addresses
	Duration computation method	Bing Maps driving durations	
	Bing Maps route type	Fastest	Recommendation: Use 'Fastest'
	Average vehicle speed	30	
3.Vehicles	Number of vehicle types	10	
4.Solution	Do the vehicles return to their depot(s)?	Yes - only once at the end	
	Time window type	Hard	
	Backhauls?	No	If activated, delivery locations must be visited before pickup locations
5.Optional - Visualization	Visualization background	Bing Maps	
	Location labels	Location IDs	
6.Solver	Warm start?	Yes	
	Show progress on the status bar?	No	
	CPU time limit (seconds)	240	Recommendation: At least 60 seconds

Appendix 8

Table 7: Client coordinates with demand

Name	Latitude (y)	Longitude (x)	Delivery amount
Depot	33.611	-7.531835	0
Customer 1	33.606	-7.627228	4
Customer 2	33.601	-7.631156	0.5
Customer 3	33.61	-7.51199	1
Customer 4	33.599	-7.488532	1.5
Customer 5	33.7	-7.395251	1.2
Customer 6	33.706	-7.34842	8
Customer 7	33.632	-7.434778	6.4
Customer 8	33.517	-7.654293	5.2
Customer 9	33.53	-7.640557	0.6
Customer 10	33.531	-7.674477	16
Customer 11	33.54	-7.675986	4.1
Customer 12	33.454	-7.644051	6.3
Customer 13	33.476	-7.606075	1.8
Customer 14	33.776	-7.158984	8.7
Customer 15	33.554	-7.481959	4.5
Customer 16	33.377	-7.568869	9.3
Customer 17	33.271	-7.580222	2.8
Customer 18	33.593	-7.615155	2.6

Appendix 9:

Table 8: Vehicle Set up

Starting depot	Vehicle type	Capacity	Return depot	Number of vehicles
Depot	Dobel box	8.4	Depot	12
	Wingbox	18	Depot	2

Appendix 10